Cepat Mahir Visual Basic .NET

M. Choirul Amri
choirul@bsmdaemon.com

Bab 3
Bekerja dengan Variabel dan Data

Seorang software developer tidak dapat mengelak untuk tidak menggunakan variabel dan data. Mau tidak mau anda harus menggunakannya. Pemahaman yang benar tentang variabel baik dalam hal scope, lifetime, dan type nya akan sangat berguna untuk melahirkan sebuah aplikasi handal, efisien, dan cepat.

Mengapa sebuah variabel harus didefinisikan dan dibuat ? Karena komputer menggunakan memory untuk menampung sementara data yang akan diproses. Ketika anda akan melakukan sebuah perhitungan, maka lebih efisien apabila anda membuat variabel-variabel untuk menampung formula perhitungan tersebut. Anda dapat saja tidak menyimpannya dalam variabel dan langsung memasukkan nilainya dalam sebuah perhitungan, dan akibatnya komputer harus meminta input dari pengguna untuk setiap nilai yang akan dihitung.

3.1 Penggunaan Variabel

Untuk dapat memakai sebuah variabel maka anda harus mendeklarasikannya terlebih dahulu. Dalam bahasa VB6 anda dapat saja menggunakan sebuah variabel tanpa membuat deklarasi meskipun hal tersebut tidak direkomendasikan dan sangat tidak efisien bagi sebuah aplikasi.

Deklarasi Variabel

Tujuan pendeklarasian variabel adalah agar komputer mengetahui dengan pasti type data yang akan digunakan dalam variabel tersebut serta scopenya. Dengan demikian komputer dapat langsung mengeksekusi sebuah variabel tanpa memeriksa lagi type datanya. Sebuah variabel harus memiliki nama, type data, scope, dan value. Berikut adalah contoh deklarasi variabel :

```
Dim sNama As String
Dim dGajiPokok As Decimal
Dim dTunjangan As Decimal
Dim dGajiTotal As Decimal
Dim dPajak As Decimal

sNama = "Anto"
dGajiPokok = 600000
dTunjangan = 150000
```

Dideklarasikan 5 variabel masing-masing dengan type string (1 variabel) dan decimal (4 variabel). Selanjutnya pada tiap variabel tersebut masing-masing diisikan nilainya. Pengisian nilai variabel tersebut harus sesuai dengan type data yang telah ditetapkan. Misalnya saja anda tidak dapat mengisi variabel `dGajiPokok dengan "Anto", karena` variabel tersebut telah didefinisikan sebagai decimal yang harus berisi angka dan bukan string.

Anda juga dapat mendeklarasikan variabel dan langsung memberikan nilainya pada saat yang sama dalam satu baris kode. Teknik ini merupakan feature baru VB .NET.

```
Dim dGajiPokok As Decimal = 600000
```

Selain itu anda juga dapat mendeklarasikan beberapa variabel sekaligus dalam satu baris dan mendefinisikan type datanya secara bersamaan.

```
Dim sUmur, sTinggi, sGaji As Single
```

Ketiga variabe tersebut memiliki type data sama yaitu Single. Cara ini memudahkan pendeklarasian variabel daripada harus mengulang deklarasi dalam tiga baris.

Melakukan Perhitungan

Selanjutnya anda dapat melakukan perhitungan tertentu dengan menggunakan variabel tersebut sebagai komponen formula sebagai berikut :

```
dGajiTotal = dGajiPokok + dTunjangan
dPajak = 0.1 * dGajiTotal
```

Segala Sesuatu adalah OBYEK !

Salah satu mantra baru dalam .NET programming adalah cara kita memandang terhadap obyek. Segala sesuatu adalah obyek, begitulah pedoman yang harus anda pegang. Dalam teknik VB6 anda mengenal konsep Object Oriented Programming (OOP) melalui Class dan konsep COM. Namun dalam .NET segala sesuatu merupakan obyek, termasuk variabel juga merupakan obyek.

Karenanya suatu varabel juga memiliki berbagai sifat yang diwujudkan dalam property, event, dan function. Sifat-sifat tersebut dapat diwariskan ke obyek lain, atau suatu variabel dapat mewarisi sifat dari obyek di atasnya.

Sebagai contoh anda dapat melakukan perhitungan besarnya dGajiTotal dan dPajak di atas dengan memanfaatkan fasilitas OOP yang terdapat dalam variabel tersebut. Gunakan keyword untuk melakukan perhitungan sebagai berikut:

```
dGajiTotal = dGajiTotal.Add(dGajiPokok, dTunjangan)
dPajak = dGajiTotal.Multiply(0.1, dGajiTotal)
```

Anda menggunakan function Add dan Multiply sebagai pengganti perhitungan sebelumnya. Argumen yang digunakan adalah dGajiPokok, dTunjangan dan dGajiTotal.

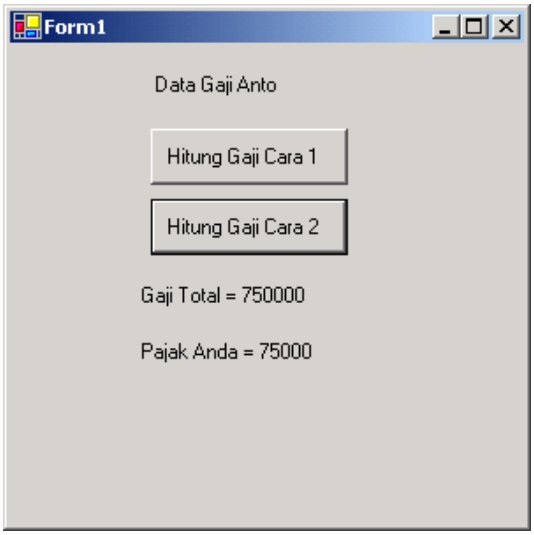
Menampilkan Hasil Perhitungan

Hasil perhitungan tersebut ditampilkan ke dalam dua buah label sebagai berikut :

```
Label1.Text = "Gaji Total = " & dGajiTotal.ToString
Label2.Text = "Pajak Anda = " & dPajak.ToString
```

Keyword ToString yang mengikuti setiap variabel bertujuan untuk mengkonversikan hasil perhitungan yang semula bertype decimal menjadi string.

Anda akan melihat bahwa perhitungan yang anda lakukan dengan cara pertama akan sama hasilnya dengan menggunakan cara kedua yang memanfaatkan built in function dalam sebuah variabel.



3.2 Ruang Lingkup dan Type Variabel

Sebuah variabel memiliki ruang lingkup (scope) tertentu, tergantung dengan cara bagaimana variabel tersebut dideklarasikan. Terdapat 3 macam scope suatu varibel :

1. Procedure level / local scope
2. Module level
3. Variabel Public

Masing-masing scope tersebut dapat dijelaskan sebagai berikut :

Procedure level / local scope

Apabila suatu variabel dideklarasikan dalam suatu prosedur tertentu, maka variabel tersebut hanya dapat diakses dan berlaku untuk prosedur tersebut saja. Perhatikan contoh kode berikut :

```
Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Button3.Click

    Dim sPesan As String
    sPesan = "tes Variabel local"

    MessageBox.Show(sPesan, "Variabel local", MessageBoxButtons.OK, _
    messageBoxIcon.Information)
End Sub
```

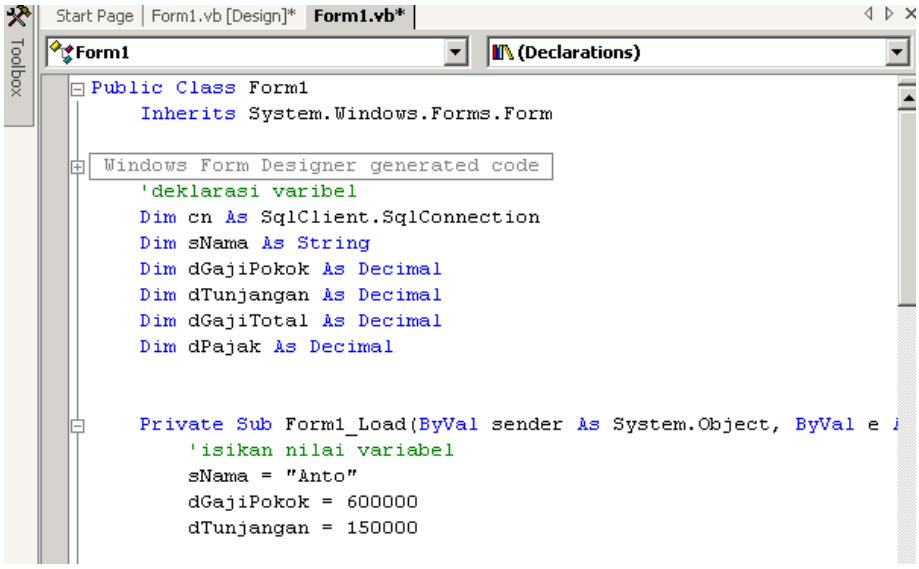
Variabel sPesan dideklarasikan di dalam prosedur Button3_Click sehingga hanya berlaku di dalam prosedur tersebut saja. sPesan tidak dapat diakses dari luar Button3_Click. Apabila anda menggunakan variabel yang hanya dipakai dalam suatu prosedur tertentu maka sebaiknya anda menggunakan jenis variabel ini.

Pengertian prosedur di sini menyangkut function, event, dan properti, sehingga tidak terbatas pada event saja sebagaimana dicontohkan di atas.

Karena variabel ini scope nya lokal untuk prosedur tertentu saja maka nama variabel tersebut hanya berlaku di dalam prosedur dimana variabel tersebut dideklarasikan. Misalkan anda memiliki function bernama A dan memiliki variabel bernama sNama. Kemudian anda memiliki sebuah event B dan memiliki variabel bernama sama yaitu sNama. Kedua variabel tersebut tidak akan saling berhubungan dan tetap terpisah nilainya karena dideklarasikan secara lokal di dalam prosedur masing-masing.

Module Level

Anda dapat membuat suatu variabel yang dapat diakes dari prosedur manapun dalam suatu file. Misalkan anda ingin mendeklarasikan suatu string koneksi yang akan dipakai terus menerus di dalam suatu module. Maka anda harus mendeklarasikan variabel tersebut di bagian deklarasi module sebagaimana contoh berikut :



Terlihat ada 6 variabel yang dideklarasikan di bagian Declaration sebuah form bernama Form1. Selanjutnya variabel yang dideklarasikan di bagian ini akan dipakai untuk melakukan berbagai perhitungan di tiap prosedur. Anda dapat membuka source code yang disertakan untuk meneliti lebih jauh bagaimana variabel tersebut dipakai.

Anda harus berhati-hati menggunakan variabel jenis ini, karena nilai suatu variabel akan terus berubah mengikuti perlakuan di setiap prosedur yang mengaksesnya. Sebaiknya anda hanya menggunakannya apabila memang benar-benar diperlukan. Untuk mempermudah menelusuri error dan maintenance aplikasi anda dapat menggunakan variabel dengan scope local.

Variabel jenis ini hanya berlaku untuk module dimana variabel tersebut dideklarasikan. Variabel ini tidak dapat diakses dari module lain meskipun berada dalam sebuah Project aplikasi yang sama. Apabila anda membutuhkan variabel yang dapat diakses dari module lain maka dapat digunakan Variabel Public/Global

Variabel Public/Global

Cara pendeklarasian variabel jenis ini hampir sama dengan jenis module level dengan menambahkan kata Public sebagai pengganti keyword Dim. Variabel ini dapat diakses dari module lain, dan bahkan dapat diakses oleh Project lain selama Project tersebut membuat reference ke Project dimana variabel tersebut dideklarasikan.

Misalkan anda memiliki Class yang digunakan untuk membuka dan menutup koneksi database. Maka anda dapat mendefinisikan variabel untuk koneksi sebagai Public sehingga form, Class, maupun Project lain dapat mengakses variabel tersebut dan melakukan koneksi ke database dengan memanfaatkan Class tersebut.

Pada contoh deklarasi di atas variabel cn dirubah deklarasinya menjadi code berikut :

```
Public cn As SqlClient.SqlConnection
```

Anda akan banyak menggunakan variabel jenis ini pada saat mempelajari teknik reference dalam mengakses berbagai komponen yang terdapat di VB.NET maupun yang anda buat sendiri.

Type Data Pada Variabel

Setiap variabel harus dideklarasikan type datanya sehingga VB mengalokasikan sumber daya yang lebih efisien untuk variabel tersebut. Anda harus memahami dengan baik tiap jenis type data dan kapan type data tertentu digunakan.

Misalnya anda menggunakan type data String untuk menyimpan nama seseorang, dan menggunakan Byte untuk menyimpan umurnya. Anda harus memilih type data dengan ukuran dan akurasi paling efisien dan cocok sesuai tujuan anda.

Contoh lain bila anda ingin melakukan kalkulasi dengan presisi tinggi, maka dapat digunakan type data Double, sedangkan untuk menyimpan umur seseorang yang tidak mungkin lebih dari 100 tahun maka digunakan type Byte.

Tabel berikut merupakan daftar type data yang dapat anda gunakan lengkap dengan ukuran dan scopenya:

Type Data	Ukuran Memori	Nilai Default	Cakupan Nilai
Boolean	4	False	True atau False
Byte	1	0	0 s/d 255
Char	2	Char(0)	0 s/d 65,535
Date	8	01/01/0001 12:00:00AM	January 1, 1 CE s/d December 31, 9999
Decimal	12	0D	+/- 9,228,162,514,264,337,593,543, 950,335 nilai bukan nol terkecil : +/-.0000000000000000000000000001
Single	4	0.0	3.402823E38 s/d -1.401298E-45 untuk nilai negatif; 1.401298E-45 s/d 3.402823E38 untuk nilai positif
Double	8	0.0	1.79769313486231E308 s/d -4.94065645841247E-324 untuk nilai negatif ; 4.94065645841247E-324 s/d 1.79769313486232E308 untuk nilai positif
Integer	4	0	- 2,147,483,648 s/d 2,147,483,647
Short	2	0	-32,768 s/d 32,767
Long	8	0	- 9,223,372,036,854,775,808 s/d 9,223,372,036,854,775,807

Berdasarkan tabel diatas anda dapat memilih penggunaan type data yang paling sesuai dengan kebutuhan dalam sebuah variabel.

Prinsip utama yang perlu diperhatikan dalam pemilihan adalah jangan menggunakan tipe data yang yang ukurannya berlebihan dibanding kebutuhan anda. Sesuaikan pula dengan presisi yang dikehendaki dalam aplikasi.

Misalnya untuk membuat variabel yang berisi jumlah anak dalam keluarga, anda menggunakan type Integer. Tentunya ini tidak sesuai karena type Integer berukuran 4 byte dalam memori, sedangkan anda tidak memerlukan nilai sampai dengan maksimum 2,147,483,647. Anda dapat menggunakan type data Byte dengan nilai maksimum 255 dan hanya membutuhkan 1 byte memori. Tentunya sesuatu yang wajar apabila jumlah anak dalam keluarga tidak melebihi angka 255.

Dua contoh perhitungan berikut menampilkan hasil yang berbeda dari suatu perhitungan yang diakibatkan penggunaan type data berbeda.

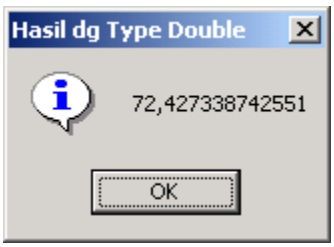
```
Dim sLuas As Single
Dim sPanjang As Single = 7.5689782
Dim sLebar As Single = 9.568972

'Hitung luas dengan tipe data single
sLuas = sPanjang * sLebar

'tampilkan di message box
MessageBox.Show(sLuas.ToString, "Hasil dg Type Single", _
MessageBoxButtons.OK, MessageBoxIcon.Information)
```

Apabila kode tersebut dieksekusi maka tampil hasil perhitugan sebagai berikut :

Tetapi apabila variabel sLuas diganti menjadi bertype Double :

```
Dim sLuas As Double
```

Maka hasil perhitungannya menjadi sebagai berikut :



Perbedaan tersebut terjadi karena type data Single dan Double memiliki tingkat presisi yang berbeda. Contoh tersebut memberikan gambaran kepada anda bagaimana pemilihan suatu variabel menjadi sesuatu yang sangat mendasar dalam sebuah aplikasi.

Type Data Untuk Semua

Telah dijelaskan dalam bagian pendahuluan kuliah berseri ini bahwa dalam Visual Studio .NET terdapat beberapa bahasa yang mendukung pemrograman .NET. Bahasa tersebut adalah VB, C#, C++, dan J#. Selain itu terdapat pula beberapa third party language yang juga mendukung pemrograman di lingkungan .NET.

Karena semua bahasa tersebut mengakses .NET Framework yang sama maka tidak terdapat perbedaan type data dalam tiap-tiap bahasa. Ini berbeda dengan kondisi sebelumnya dimana type data dalam VB berbeda dengan VC++ dan masing-masing harus dipertimbangkan kompatibilitasnya.

Kompatibilitas data dan aplikasi dalam .NET lebih terjamin karena menggunakan library yang sama dengan type data sama pula. Ini merupakan kabar gembira bagi sebuah tim pengembang aplikasi yang terdiri dari berbagai programmer dengan keahlian bahasa yang berbeda.

Konversi Variabel

Pada saat tertentu mungkin anda memerlukan konversi dari satu type data ke type lain. VB .NET menyediakan fungsi `CType` untuk mengkonversikan variabel. Contoh berikut mengkonversikan variabel umur yang semula bertipe String menjadi Single.

```
Dim sUmur As String = 5.5
Dim bUmur As Single = CType(sUmur, Single)

MessageBox.Show(bUmur, "Hasil Konversi", MessageBoxButtons.OK, _
MessageBoxIcon.Information)
```

Inti konversi adalah memasukkan obyek yang akan dikonversi sebagai argumen dari `CType` dan menentukan variabel tujuan konversinya.

System Namespace

Dalam .NET sekumpulan kelas library yang memiliki fungsi tertentu disebut dengan namespace. Karena .NET mendukung implementasi konsep OOP maka semua obyek dalam setiap bahasa merupakan turunan dari Namespace tersebut. Type data yang telah dijelaskan sebelumnya sebenarnya diturunkan dari System Namespace, yang merupakan root namespace dalam sistem .NET.

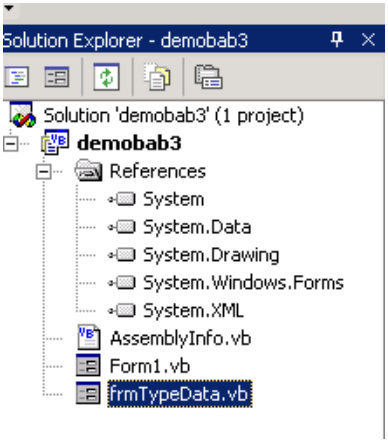
Misalnya type data Decimal, ternyata diturunkan dari System Namespace yaitu System.Decimal runtime structure. Demikian pula dengan type data yang lain, selalu diturunkan dari namespace System tersebut.

Dengan demikian anda memiliki cara lain dalam mendeklarasikan variabel dan melakukan perhitungan sebagai berikut :

```
Dim dGaji As System.Decimal = 1000000
Dim sPajak As System.Decimal = 0.1

Dim dGajiTerima As System.Double = dGaji * (1 + sPajak)

MessageBox.Show(dGajiTerima.ToString, "Hasil Hitung", _
MessageBoxButtons.OK)
```

Mungkin anda bertanya, mengapa pendeklarasian Decimal tetap dapat dibenarkan daripada secara lengkap menyebutkan System.Decimal ? Jawabannya adalah karena VB secara default telah menyertakan namespace System sebagai reference library dalam setiap Project. Karena reference tersebut telah dibuat maka tidak menjadi halangan apabila anda langsung menyebutkan nama type datanya saja. Konsep ini berlaku untuk semua namespace dalam .NET.

Apabila anda perhatikan menu Solution Explorer maka terlihat beberapa namespace telah direferensikan secara default, yaitu System, System.Data, System.Drawing, dan seterusnya.



Anda dapat menambahkan refrensi tersebut sesuai dengan kebuthan dan jenis library yang akan diakses. Pada bab-bab selanjutnya anda akan mempelajari fungsi apa saja yang tersedia dalam setiap namespace dan bagaimana cara penggunaannya.

3.3 Constant

Apabila sebuah variabel selalu memiliki nilai tetap dan tidak berubah-ubah di sepanjang aplikasi maka lebih baik bila anda mendefinisikannya sebagai constant. Misalnya dalam sebuah aplikasi matematika yang memiliki variabel phi, dimana phi bernilai 3.14 yang digunakan dalam perhitungan luas lingkaran dan volume tabung.

Menetapkan sebuah variabel tetap sebagai constant memiliki keuntungan karena constant dieksekusi lebih cepat daripada variabel. Ini berarti peningkatan performa aplikasi yang dibangun.

Deklarasi constant sama dengan deklarasi variabel, dengan menambahkan keyword Constant di depan nama variabel.

```
Const phi As single = 3.14
Dim sRadius As single = 20
Dim sKeliling As single

'melakukan perhitungan dengan constant
sKeliling = 2 * phi * sRadius
```

Semua jenis type data, scope, dan tata cara deklarasi yang berlaku pada variabel berlaku pula untuk constant. Perbedaanya adalah apabila variabel dapat berubah-ubah nilainya, sedangkan constant selalu bernilai tetap.